

Teóricos

Ejercicio 2 - System Programming - (10 puntos)

1. A continuación se tiene las entradas de una TLB correspondientes a una misma tarea que ejecuta en un sistema basado en un procesador x86. Las entradas están en el orden en el que se han ido generando las traducciones. Para dicha tarea el registro CR3 = 0x000E4000, y las tablas de página correspondientes a las direcciones en uso están inmediatamente contiguas al DTP, en el orden en el que se han ido requiriendo.

#	Nro.Pag.	Descriptor	Ctrl
1	7C047	0EF00121	ccc
2	7EEF0	1EF01067	ccc
3	EC004	001F0163	ccc
4	EC005	001F7123	ccc
5	46104	1F011005	ccc
6	46109	1F010027	ccc

Se pide:

- (a) Para las entradas 1 y 2 de la TLB, escribir el contenido de sus correspondientes descriptores en cada nivel de la jerarquía de tablas de traducción a direcciones físicas, considerando que se trata de las dos primeras páginas requeridas al ejecutar la tarea.
- (b) Especificar para cada PDE que valor deben tener los bits U/S y R/W, para ser consistentes con el contenido de la TLB. Respuestas posibles en cada caso: 0, 1, o X (indistinto).
- (c) Suponiendo que las seis entradas están siendo utilizadas por una misma tarea, y por cada task switch se modifica CR3. ¿Que ocurre con cada una al momento del task switch? ¿Cual es el tratamiento para aquellas que se han modificado?
- (d) ¿Cual es la función dentro de la TLB de los bits identificados genéricamente como Ctrl?

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Address of page directory ¹																				Ignored				P C D	P W T	Ignored				CR3				
Bits 31:22 of address of 4MB page frame										Reserved (must be 0)				Bits 39:32 of address ²		P A T	Ignored	G	1	D	A	P C D	P W T	U / S	R / W	1	PDE: 4MB page							
Address of page table																				Ignored				Q	I g n	A	P C D	P W T	U / S	R / W	1	PDE: page table		
Ignored																										Q	PDE: not present							
Address of 4KB page frame																				Ignored				G	P A T	D	A	P C D	P W T	U / S	R / W	1	PTE: 4KB page	
Ignored																										Q	PTE: not present							

2. ¿Porqué un sistema que utilice dos o más niveles de privilegio diferentes debe usar una TSS aun cuando la conmutación de tareas es manual?

Ejercicio 3 - Micro Arquitectura - (15 puntos)

1. Se tiene un sistema SMP. Cada procesador tiene su propio controlador Cache. Utiliza para mantener coherente los sub sistemas de memoria cache y la DRAM el protocolo MESI
 - (a) ¿Cuál es el recurso de hardware mediante el cual cada Controlador Cache detecta las transacciones que los demás Cores cursan con la memoria del sistema (DRAM)?. Indicar el nombre del recurso, y a que líneas del bus se conecta.
 - (b) Explicar si M es un estado preciso o impreciso. Justificar
 - (c) En caso que otro procesador inicie una transferencia de lectura sobre un dato que éste procesador tiene en una línea, cuyo estado es **M**, se desea saber:
 - i. ¿Están coherentes las diferentes copias del dato requerido?. Justificar.
 - ii. ¿Cómo se asegura la coherencia de la operación?. Justificar. Detallar el handshake de hardware describiendo las líneas involucradas si corresponde.
 - iii. ¿Cuál es la política de escritura aplicada por el procesador antes de la transacción analizada y después de la misma?
2. Considerar el siguiente código:

```

1  #define count 1024
2
3  section .data
4  buffer:      dq  0x4523, 0xFFFF5666723, ... ; 'count' values
5  result:      resq 2
6
7  section .text
8  mov     RSI,buffer
9  mov     RCX,count
10 mov     RDI,result
11 lods
12 add     RBX,RAX ; Estado luego de ejecutada esta instrucción!
13 mov     RAX,RBX
14 idiv    RCX
15 mov     [RDI],RAX
16 mov     [RDI+8],RDX

```

Se pide:

- (a) Enumerar los pares de instrucciones con hazards debidos a que un procesador de Intel ejecuta fuera de orden. Explicar que significa cada uno de los hazards.
- (b) Ubicar los valores apropiados en cada TAG, VALUE, y V, para cada elemento involucrado en el algoritmo anterior para el modelo de Tomasulo que se representa a continuación, cuando se termina de ejecutar la instrucción de la línea de código 12 del listado anterior. Se pretende tener el estado de máquina terminada dicha instrucción.

1)

 $CR3 = 0x000E4000$ DIRECCIÓN DEL PAGE DIRECTORY $PT1 = CR3 + 0x1000 = 0x000E5000$ $PT2 = CR3 + 0x2000 = 0x000E6000$ DIRECCIONES DE
LAS PAGE TABLESASUMO $CRO.WP = 1$

ENTRADA TLB #1

DESARMAMOS EL NÚMERO DE PÁGINA EN LA TLB

 $0x7C047 \Rightarrow \#PDE = 0x1F0$ PAGE DIRECTORY ENTRY INDEX $\#PTE = 0x047$ PAGE TABLE ENTRY INDEX

DESARMAMOS EL DESCRIPTOR CACHEADO EN LA TLB

 $0x0EF00121 \Rightarrow$ BASE FÍSICA = $0x0EF00$

ATRIBUTOS COMBINADOS:

GLOBAL=1, PAT=0, DIRTY=0, ACCESSED=1,

PCD=0, PWT=0, U/S=0, R/W=0

PDE #0x1F0

ADDRESS = $PT1 \gg 12 = 0x000E5$

ACCESSED=1, PCD=0, PWT=0, U/S=X, R/W=X

NO IMPORTAN PORQUE GANAN LOS
DE LA PTE.

PTE #0x047

ADDRESS = $0x0EF00$

GLOBAL=1, PAT=0, DIRTY=0, ACCESSED=1,

PCD=0, PWT=0, U/S=0, R/W=0

ENTRADA TLB #2

$0x7EEF0 \Rightarrow \#PDE = 0x1FB$

$\#PTE = 0x2F0$

$0x1EF01067 \Rightarrow \text{BASE FÍSICA} = 0x1EF01$

ATRIBUTOS COMBINADOS:

GLOBAL=0, PAT=0, DIRTY=1, ACCESSED=1,

PCD=0, PWT=0, U/S=1, R/W=1

PDE #0x1FB

ADDRESS = PT2 \gg 12 = $0x000E6$

ACCESSED=1, PCD=0, PWT=0, U/S=1, R/W=1

PTE #0x2F0

ADDRESS = $0x1EF01$

GLOBAL=0, PAT=0, DIRTY=1, ACCESSED=1,

PCD=0, PWT=0, U/S=1, R/W=1

AL REALIZAR EL TASK SWITCH Y CAMBIAR EL CR3, NECESITAMOS FLUSHEAR (INVALIDAR) LA TLB ENTERA PORQUE EL MICRO NO TIENE FORMA DE SABER QUE TRADUCCIONES SIGUEN VIGENTES EN EL NUEVO ESQUEMA DE PAGUACION.

NO OBSTANTE, LAS TRADUCCIONES CACHEADAS COMO GLOBALES PERMANECEN EN LA TLB. ESTO PERMITE MANTENER LAS TRADUCCIONES PARA EL IDENTITY MAPPING DEL KERNEL. DE SER NECESARIO, SE PUEDEN INVALIDAR MANUALMENTE.

ENTONCES, AL CAMBIAR EL CR3, LAS GLOBALES QUEDAN EN LA TLB, EL RESTO SE INVALIDA.

DE LAS QUE SE INVALIDAN, TODAS LAS PAGINAS MARCADAS COMO DIRTY PRIMERO DEBEN SER ESCRITAS EN DISCO, PUES SINO SE PERDERIRIAN DATOS.

EL SISTEMA OPERATIVO TAMBIEN PUEDE ESCRIBIR A DISCO LAS PAGINAS DIRTY DURANTE MOMENTOS DE BAJA UTILIZACION DEL SISTEMA. ~~SI BIEN LA TRADUCCION VIRTUAL A FISICA QUEDA~~

~~CACHEADA EN LA TLB, LOS PAGOS DIRTY Y ACCESOS SON~~
~~IDENTIFICADOS SIMULTANEAMENTE EN LA TLB, E INVALIDAN~~
~~LA TLB~~

LOS BITS IDENTIFICADOS COMO CTL SON TÍPICOS BITS DE CONTROL PARA UNA CACHE:

- VALID: INDICA SI LA TRADUCCIÓN ES VÁLIDA O NO
- LRU: LEAST RECENTLY USED, PERMITE IMPLEMENTAR UNA POLÍTICA DE DESALOJO DE LA TRADUCCIÓN MÁS VIEJA SIN USAR CUANDO SE LLENA LA TLB.

~~2)~~

~~NECESITAMOS CONFIGURAR SI O SI UNA TSS PARA PODER GUARDAR~~
~~LA DISEÑO~~

2)

POR MÁS QUE IMPLEMENTEMOS UNA CONMUTACIÓN DE TAREAS CON UNA ESTRUCTURA CUSTOM EN VEZ DE LA TSS Y UN MECANISMO MANUAL DE PRESERVACIÓN Y RESTAURACIÓN DEL CONTEXTO DE EJECUCIÓN, NECESITAMOS UNA TSS PARA QUE EL CPU PUEDA REALIZAR EL CAMBIO DE PRIVILEGIO AUTOMÁTICO AL ENTRAR EN UNA INTERRUPCIÓN DE EXCEPCIÓN.

PUNTUALMENTE, AL CAMBIAR DE PRIVILEGIO NECESITAMOS USAR OTRA PILA DE NIVEL 0 Y PRESERVAR LA DE NIVEL 3. EL CPU BUSCA EN LA TSS EL ESP0 Y SS0 AL CAMBIAR DE PRIVILEGIO ENTRANDO EN LA INTERRUPCIÓN.

1) A)

EL SNOOP BUS PERMITE OBSERVAR LAS TRANSACCIONES ENTRE LOS CORES Y LA MEMORIA. NO NOS IMPORTAN LOS DATOS, SOLO LAS DIRECCIONES Y SEÑALES DE CONTROL (ALGUNAS, POR EJEMPLO SI LA OPERACIÓN ES READ O WRITE) PARA PODER ACTUAR SI EL CACHE LOCAL POSEE ESA MISMA DIRECCIÓN, EN ALGÚN ESTADO PUNTUAL. EN ESENCIA, SIN ESTO SERÍA IMPOSIBLE MANTENER LA COHERENCIA PORQUE LOS CORES NO SABRÍAN QUÉ ESTÁN HACIENDO LOS OTROS CORES.

1) B)

EL ESTADO M (MODIFIED) ES PRECISO. SUPONGAMOS QUE 2 CORES TRABAJAN CON MESI. EL CORE 1 LEE UN DATO Y AL NO RECIBIR UN ~~EXCLUSIVE~~ ^{SHARED}, MARCA SU ENTRADA COMO EXCLUSIVE. LUEGO EL CORE 2 LEE LA MISMA DIRECCIÓN, Y POR EL SNOOP BUS EL CORE 1 SE ENTERA Y AVISA AL CORE 2 PARA QUE AMBOS PONGAN ESE DATO (LÍNEA DE CACHE) EN SHARED.

MÁS TARDE, EL CORE 1 MODIFICA EL DATO, Y POR EL SNOOP BUS NUEVAMENTE EL CORE 2 SE ENTERA E INVALIDA EL DATO. EL CORE 1 LO MARCA COMO MODIFIED. SI NO HABÍA OTRO CORE CON EL MISMO DATO NO PASA NADA, EL CORE 1 IGUAL LO MARCA COMO M.

SI EL DATO LLEGO A M, ESE CORE ES EL ÚNICO QUE POSEE EL DATO. POR ESO ES PRECISO, AL IGUAL QUE EXCLUSIVE.

1) c)

- NO ESTÁN COHERENTES. EL CORE QUE TIENE LA LÍNEA EN M POSEE UN VALOR QUE DIFIERE DE MEMORIA PUES AÚN NO LO ESCRIBIÓ EN MEMORIA (POR ESO M).
- ESCUCHA POR EL SNOOP BUS QUE OTRO CORE QUIERE LEER EL DATO. ENVÍA UN RFO PARA QUE EL OTRO CORE INVALIDE LA LÍNEA E INICIA EL COPY BACK DE LA LÍNEA MODIFICADA.
- NO ME QUEDA CLARA LA PREGUNTA. PERO CREO QUE ES ESTO: CUANDO LE MANDA EL RFO Y HACE EL COPY BACK, EL CORE QUE QUERÍA LEER TOMA EL NUEVO VALOR POR EL BUS Y AMBOS CORES MARCAN ESA LÍNEA COMO SHARED.

2)E) AL ENTRAR LA INSTRUCCIÓN:

MOV RAX, RBX

SE PRODUCE STALL PORQUE LA UNIDAD DE EJECUCIÓN
LOAD/STORE NO TIENE ESPACIO EN SU RESERVATION
STATION.

2)A)

WAR 11-8

WAR = WRITE AFTER READ ✓

RAW 12-11

RAW = READ AFTER WRITE ✓

WAW 13-11

WAW = WRITE AFTER WRITE ✓

RAW 13-12

LEO/ESCRIBO ANTES QUE UNA ✓

RAW 14-9

OPERACIÓN ANTERIOR TERMINE DE ✓

WAW 14-13

ESCRIBIR. ✓

WAW 14-11

SE ARREGLA ^{RAW} CON EL REORDER BUFFER

COMITANDO / RETIRANDO LOS RESULTADOS

A LA ARQ, EN EL ORDEN ORIGINAL
DE LAS INSTRUCCIONES. ✓